Trabajo Fin de Grado
Grado en Ingeniería de Tecnologías Industriales

Sistema de adquisición, gestión y supervisión de datos en tiempo real de un cuadro de baja tensión

Autor: Fernando Arteaga Carrillo
Tutor: Ignacio Alvarado Aldea

Dpto. Ingeniería de Sistemas y Automática
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2019

Trabajo Fin de Grado
Grado en Ingeniería de Tecnologías Industriales

Sistema de adquisición, gestión y supervisión de datos en tiempo real de un cuadro de baja tensión

Autor:
Fernando Arteaga Carrillo

Tutor:
Ignacio Alvarado Aldea
Profesor titular

Dpto. de Ingeniería de Sistemas y Automática
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla
Sevilla, 2019

baja tensión

Autor: Fernando Arteaga Carrillo

Tutor: Ignacio Alvarado Aldea

El tribunal nombrado para juzgar el Proyecto arriba indicado, compuesto por los siguientes miembros:

Presidente:

Vocales:

Secretario:

Acuerdan otorgarle la calificación de:

Sevilla, 2019

El Secretario del Tribunal

A todo el que haya leído estas palabras.

Gracias.

Agradecimientos

A mi madre, por enseñarme a ser mejor persona. A mi padre, por enseñarme a ser curioso. A mi hermano, por enseñarme ser constante. A mi hermana, por enseñarme a que nunca se debe perder la esperanza. A mis amigos, porque también sois mi familia. Este documento no existiría sin vosotros.

Fernando Arteaga Carrillo

Sevilla, 2019

Resumen

En el contenido de este proyecto se recoge un trabajo práctico consistente en la recogida de datos en tiempo real de los distintos parámetros eléctricos evaluables en un cuadro general de baja tensión, a través de unos dispositivos llamados "Powertags" (Schneider Electric) que se comunican a través de protocolo modbus wireless con su maestro, un dispositivo inteligente dotado con tecnología wireless, ethernet y serial, así como E/S distribuidas y una entrada analógica llamado "Smartlink SI B Ethernet (Schneider Electric)". A través de este dispositivo se controlarán las líneas de alimentación a las mesas de trabajo, las líneas de climatización de la nave industrial, así como la iluminación exterior y las líneas de extractores.

A través de un PLC "TM241CE24R" (Schneider Electric) se recavarán los datos a través de Modbus TCP programado con el software SoMachine (Schneider Electric), creando también una interfaz web a modo de Scada, con distintas páginas y gráficos que mostrarán la información y donde se podrá interactuar con ella. El sistema realizará una gestión de los datos de forma que irá escribiendo en archivos con extensión "csv" la información y se enviará a final de cada mes vía email.

Finalmente se desarrolla de forma adicional una aplicación de Realidad Aumentada con el software Augmented Operator Advisor y Node-Red, dónde se verán datos en tiempo real en un dispositivo móvil gracias al servidor OPC UA del PLC.

Abstract

The content of this project includes practical work consisting of the collection of real-time data of the different electrical parameters evaluable in a general low-voltage circuit, through devices called "Powertags" (Schneider Electric) that communicate through modbus wireless protocol with its master, an intelligent device equipped with wireless, ethernet and serial technology, as well as distributed I / O and an analog input called "Smartlink SI B Ethernet (Schneider Electric)". Through this device, the power lines to the work tables, the air conditioning lines of the industrial building, as well as the exterior lighting and the exhaust lines will be controlled.

Through a "TM241CE24R" PLC (Schneider Electric) the data will be collected through Modbus TCP programmed with SoMachine (Schneider Electric) software, also creating a web interface as a Scada, with different pages and graphics that will display the information and where you can interact with her. The system will manage the data so that the information will be written in files with the extension "csv" and will be sent at the end of each month via email. Finally, an Augmented Reality application is further developed with the Augmented Operator Advisor and Node-Red software, where real-time data will be seen on a mobile device thanks to the OPC UA server of the PLC.

Índice

ÍNDICE DE FIGURAS

1 INTRODUCCIÓN

Este proyecto surge como un trabajo de mejora interna en una empresa del sector eléctrico en Sevilla, en el cual se tiene una nave industrial con un cuadro general de baja tensión con elementos de medida y control instalados en él. Se propone pues, realizar una integración de los elementos dispuestos en el cuadro en la línea de los nuevos proyectos en el ámbito de la Industria 4.0, pasando de tener elementos aislados a conectados y controlados a través de tecnologías de internet. De esta forma, se tratará de sacar el máximo partido con los elementos dispuestos y con los conocimientos de nuevas tecnologías como la realidad aumentada, teniendo control del cuadro eléctrico y atrayendo así a potenciales clientes para proyectos similares.

1.1 Contexto histórico

La nueva industria del Siglo XXI o Industria 4.0 es el último capítulo de las sucesivas revoluciones que se han ido dando en la industria a lo largo de la historia como respuesta a una necesidad. La máquina de vapor en el Siglo XVIII supuso el primer capítulo en la historia de revoluciones industriales, produciendo un cambio no solo industrial, sino social, económico y tecnológico.

Tras esto, en el siglo XIX, se produciría una necesidad de producir más rápido y con una mayor calidad, ya que la red de transporte ferroviario y la aparición de nuevos materiales como el acero, aluminio o cobre lo favorecía. Nace entonces la división de tareas y la producción en masa, siendo Henry Ford su precursor, haciendo que el coste de productos anteriormente destinados a la clase alta de la sociedad, pudiesen ser accesibles a todos.

Por último, de la necesidad de expansión y globalización de la sociedad y la industria en el siglo XX, nace Internet, concepto reciente y sobre el que se desarrolan todas las Tecnologías de Información y Comunicación, llegando así a la automatización de los procesos y por tanto a la tercera revolución industrial.

En 2011, en la Feria anual de Hannover, se propone un objetivo a nivel industrial, una aspiración: Una industria inteligente. Es la primera vez que surge el concepto de Industria 4.0 y aunque en un principio fue una iniciativa alemana, hoy en día, practicamente todos los países se han sumado a ella. Una necesidad de mejora, eficiencia y flexibilidad junto con el desarrollo de nuevas tecnologías como la Inteligencia Artificial (AI), el Big Data, el Internet de las Cosas (IoT) o los Sistemas Ciberfísicos hace que recibamos este cambio en el paradigma de la industria como la cuarta revolución industrial.

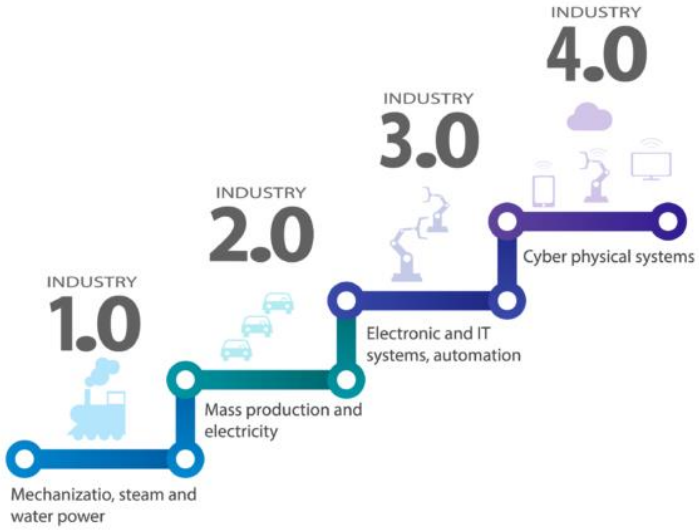


Figura 1-1. Esquema representativo de las revoluciones Industriales

1.2 Eficiencia energética e Industria 4.0

La evolución en ciernes que permitirá este tipo de industria tiene cimientos muy firmes en la optimización de procesos. Y este fenómeno no puede darse si no se considera la monitorización del consumo energético y la implementación de sistemas y procesos sostenibles. Es decir, se trata de reducir costes, no solo con el objetivo de ahorrar, sino de ofrecer soluciones inteligentes que reduzcan la demanda de recursos y de energía.

De acuerdo con la Comisión Europea habría que invertir 2 000 millones de euros anualmente para mejorar la eficiencia energética en un 25%. Sin embargo, esa inversión puede generar un ahorro de 9 000 millones de euros hacia el año 2030. Estas cifras dan medida de la significación de estas alternativas y de la importancia de apostar por una revolución tecnológica que asiente las bases de procesos más competitivos e innovadores.

1.3 Objetivo

Teniendo en cuenta el avance frenético de la industria 4.0, y la alta competitividad en el sector industrial, se propone como objetivo la mejora interna en lo que a eficiencia energética se refiere y una imagen de empresa moderna e innovadora. En este caso, el trabajo que se expone en este proyecto es un desarrollo para una empresa del sector eléctrico que realiza también la ingeniería en ciertos proyectos, por ello se tratará también como objetivo la posible compra de este proyecto adaptado a las necesidades del cliente.

1.4 Alcance

Como alcance del proyecto se tiene el control y monitorización de datos en tiempo real, así como el tratamiento de esos datos de las distintas líneas de un cuadro general de baja tensión. No se define en el alcance el diseño del cuadro, elección de Hardware específico ni elección de Software para la visualización.

En este caso el cuadro ya está montado y funcionando correctamente con el Hardware necesario para la realización del proyecto propuesto. Se aporta también por parte de la empresa los esquemas multifilares.

Por último, se realiza un estudio de cómo seguir ampliando el proyecto, utilizando tecnologías IoT con servidores Linux y realidad aumentada que harán del proyecto mucho más atractivo a clientes, ya que añadirán valor y funcionalidades muy interesantes.

2 HARDWARE

En este capítulo se describe todo el elemento físico del que se dispone para realizar el trabajo. Cabe destacar que los elementos estaban ya instalados en el momento de la realización del trabajo, por lo que se puede considerar la extrapolación a cualquier sistema similar ya existente.

2.1 Cuadro general de baja tensión

En una instalación de Baja Tensión la diferencia de potencial máxima entre dos conductores es inferior a 1000 voltios (1 kV), pero superior a 24 voltios. En el caso particular de nuestro cuadro, tenemos 200 Amperios en el embarrado principal, protegido ante sobretensiones permanentes y transitorias, que deriva en las distintas líneas destinadas a alimentar los aparatos eléctricos y electrónicos de la nave industrial.

Cada elemento de cada línea independiente está protegido con Interruptores Magnetotérmicos contra sobrecorrientes y en algunos casos, estos magnetotérmicos llevan acoplados unos relés que servirán para controlar el apagado y encendido de la línea.

2.2 Autómata

Un PLC (Programmable Logic Controller) es un ordenador usado en la industria para realizar el control de sistemas. El uso de los PLC en la industria es extenso y variado: refinería, packaging, sistemas de ventilación, control de elevadores, etc. Los PLC están compuestos por una CPU (Computer Process Unit) que se dedica a ejecutar el programa cargado en él y a monitorizar las diferentes entradas y salidas que existen en el mismo dispositivo para manipularlas con la lógica programada según la forma que se quiera controlar el sistema. Estos dispositivos están pensados para ser flexibles y robustos, ya que trabajan bajo condiciones diversas de temperatura y humedad dentro de los cuadros o máquinas dónde normalmente están alojados, pudiéndose ampliar con ciertos módulos de entradas y salidas o de comunicaciones.

El PLC que se utilizará en el proyecto será el TM241CE24R de Schneider Electric, dotato con un módulo Switch Ethernet de 4 puertos que permitirá conectar a la red de la nave industrial todos los elementos del cuadro.

Figura 2-1. Autómata TM241CE24R de Schneider Electric

Las características principales de este autómata son:

- 14 Entradas Digitales
- 6 Salidas a relé (24Vdc) o transistor(5-125Vdc o 5-250Vac)
- Protecciones en entradas y salidas
- Memoria: 8Mbytes para Programa y 64Mbytes de RAM
- 2 Puertos Serie
- 1 Puerto Ethernet (RJ-45)
- Cliente y Servidor Modbus TCP
- Ethernet
- Servidor OPC UA
- DHCP
- Servidor y Cliente FTP
- Web Server
- Cliente y Servidor SNMP
- Cliente SQL
- Memoria ampliable con Tarjeta SD hasta 32Gbytes

2.3 Smartlink SI B (Ethernet)

El Smartlink SI B (Ethernet) del fabricante Schneider Electric, es un dispositivo inteligente destinado a controlar dispositivos Modbus, tanto TCP/IP como RTU. También es posible controlar hasta 7 entradas y salidas digitales, normalmente destinadas a contactores auxiliares, y 1 entrada analógica. El dispositivo no es programable, solamente podrá controlar elementos dentro de la gama Acti9 de Smartlink, que comprende estos medidores, contadores de pulsos, contactores auxiliares y otros dispositivos destinados al control de cuadros de baja tensión, pero en este proyecto se utilizará como maestro Modbus para acceder a los datos en tiempo real de los medidores. Cabe destacar la importancia de este dispositivo dado que los medidores se comunican por ondas de radio y no es posible acceder a ellos directamente desde el PLC.

Figura 2-2. Dispositivo Smartlink SI B (Ethernet)

A continuación, se describen las características principales del Smartlink SI B (Ethernet):

- 7 Canales de Entradas y Salidas digitales
- Canal con 2 entradas analógicas
- Conector Modbus de 4 polos
- Puerto de comunicación Ethernet
- Servidor web
- Protecciones contra cortocircuitos y sobetensiones
- Servidor Modbus TCP/IP

2.4 Acti 9 PowerTag

El dispositivo Acti 9 Powertag (Schneider Electric) es un medidor de energía con tecnología wireless, fácil de instalar ya que se conecta encima o debajo del dispositivo de protección de la línea a medir, haciendo así que se ahorre espacio y tiempo de instalación, ya que son visibles en todo momento por el Smartlink.

Las medidas se realizan conforme a la norma IEC 61557-12 siendo el dispositivo un sersor de energía de clase 1, de modo que proveé datos sobre: Tensiones Fase-Fase y Fase-Neutro, Corriente por Fase, Potencias Activa y Reactiva, Potencia Aparente total y factor de potencia.

La gama PowerTag pueden medir líneas monofásicas y trifásicas hasta 630A por lo que son ideales para los cuadros de baja tensión.

Figura 2-3. Módulo de comunicación Acti 9 Powertag

3 SOFTWARE

Se expone el software utilizado para este proyecto, en este caso se hará uso de tres herramientas, la primera SoMachine se utilizará para programar el PLC, Node-Red será la herramienta que se utilizará para recibir las variables del servidor OPC en el PLC y enviarlas al servidor HTTP vinculado a la aplicación de realidad aumentada Augmented Operator Advisor.

Se ha utilizado Software gratuito, siendo el caso del Augmented Operator Advisor una versión de prueba.

3.1 SoMachine

SoMachine es el software destinado a la programación de autómatas de la gama Modicon de Schneider Electric. Este software está basado en el software de programación Codesys, añadiendo las librerias correspondientes al hardware del fabricante, haciendo que los dispositivos conectados aparezcan automáticamente.

Al igual que Codesys, SoMachine permite la programación de PLC en distintos lenguajes según la norma IEC 61131-3:

- CFC (Continous Function Chart)
- ST (Structured Text)
- LD (Ladder)
- SFC (Sequential Function Chart)

Otra característica principal de SoMachine es la compatibilidad de Librerías de Codesys, además de las importadas por Schneider Electric para sus disposivos. Esto es importante para este tipo de proyectos en materia de comunicación ya que existen numerosas librerías de codigo abierto para Codesys, por ejemplo, Oscat, donde se pueden utilizar bloques especiales de funciones para comunicar por ejemplo por protocolo MQTT.

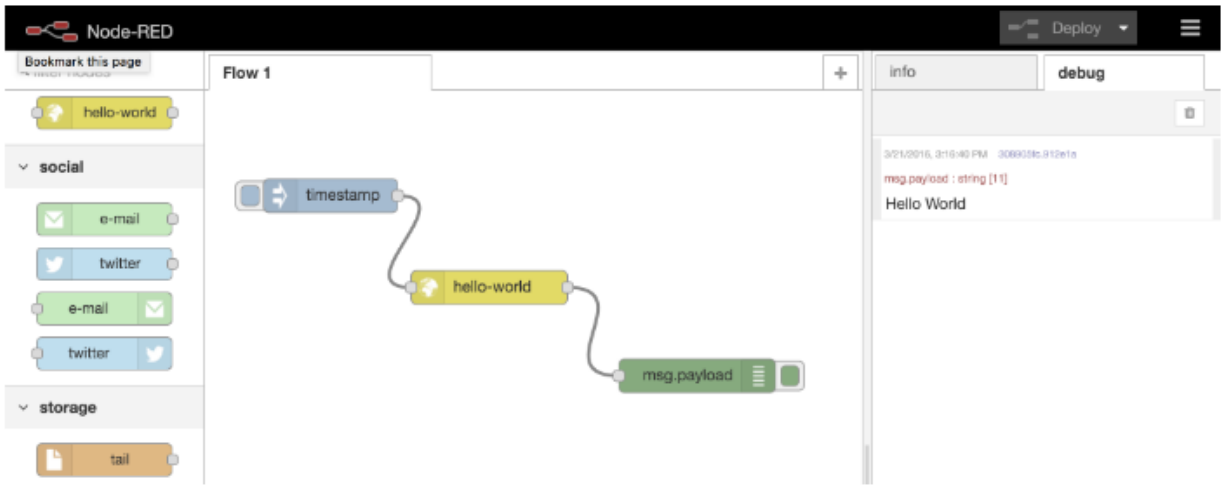
El resto de las características que definen a este Software son las siguientes:

- Posibilidad de crear un archivo de datos de texto con la funcion Datalog, con la que se creará directamente un archivo con extensión txt en la carpeta “/usr/log”.
- Se podrán crear visualizaciones con diversos objetos existentes en la aplicación, como trazas o histogramas de barras. Una vez se hayan creado estas visualizaciones se podrán cargar en el servidor de visualización web del dispositivo (si tuviera). El dispositivo utiliza un servidor http donde se carga automáticamente la visualización en formato htm y a través del puerto 8080 se puede acceder a ella.
- Permite la Configuración de Símbolo, que da la posibilidad de seleccionar directamente qué variables se van a comunicar con los diferentes dispositivos y aplicaciones dentro de un mismo proyecto, por ejemplo, un HMI.
- Es posible enviar y recibir variables en red mediante Modbus RTU directamente sobre SoMachine eligiendo las variables que se han definido en el proyecto y en caso de enviarla, definir a qué dirección y puerto se van a mandar. En caso de recibirlas, habría que proceder de forma inversa.
- Configuración sencilla del servidor OPC UA símplemente eligiendo qué variables van a poder ser visibles por un cliente, el tipo de identificador, y el puerto.

3.2 Node-Red

Node-Red es una herramienta de programación basada en un sistema de módulos de tipo Scrach que combina la simplicidad de la programación con módulos, con la posibilidad de implementar codificaciones en JavaScript.

Figura 3-1. Ejemplo de proyecto en Node-Red



Esta herramienta consiste en una Runtime basada en Node.js, accediendo al editor de la herramienta cuando se apunta a la dirección del servidor que está ejecutando Node-Red desde el explorador de internet.

Creado con el propósito de visualizar y manipular los 'topics' del protocolo de comunicaciones MQTT, rápidamente se adaptó a propósitos generales, al ser una herramienta muy potente y de código abierto. De manera muy sencilla se pueden instalar nuevos paquetes de nodos con propósitos diversos de distintos desarrolladores gracias a que los proyectos se guardan en formato JSON (JavaScript Object Notation), haciendo que se puedan importar y exportar de forma sencilla entre la comunidad de Node-Red.

3.3 Augmented Operator Advisor

Augmented Operator Advisor utiliza la tecnología denominada realidad aumentada para optimizar el funcionamiento y el mantenimiento de los emplazamientos y equipos industriales:

- El operador apunta una tableta hacia el emplazamiento o equipo a supervisar.
- EcoStruxure Augmented Operator Advisor utiliza técnicas especiales de comparación de imágenes para emparejar la escena visible del campo de visión de la cámara de la tableta con fotografías almacenadas previamente de la misma escena.
- Cuando se logra una coincidencia, los marcadores denominados puntos de interés se superponen sobre la escena en tiempo real visible en la tableta.

- El operador pulsa los marcadores de punto de interés para mostrar información. Se puede visualizar

una gran variedad de información, que incluye:

- Variables de proceso
- Valores extraídos de una base de datos SQL
- Documentos, hojas de instrucciones o diagramas de cableado
- Páginas web
- Vídeos
- Audio
- Procedimientos

Figura 3-2. Ejemplo de runtime Augmented Operator Advisor en dispositivo móvil.

La aplicación de desarrollo de proyectos de realidad aumentada es accesible desde la web de Schneider Electric, con una cuenta de partner del fabricante se puede acceder a la versión de prueba del Software. Una vez en el entorno de desarrollo del proyecto, se procede a cargar la imagen o tag que posteriormente será reconocida por la cámara de nuestro dispositivo móvil, a continuación, se modifica el área, en el que se podrán introducir los elementos visibles en realidad aumentada: variables, subescenas, imágenes, texto, etc.

4 PROTOCOLOS DE COMUNICACIÓN

Un protocolo de comunicación es un conjunto de reglas que permiten la transferencia e intercambio de datos entre los distintos dispositivos que conforman una red. Estos han tenido un proceso de evolución gradual a medida que la tecnología electrónica ha avanzado y muy en especial en lo que se refiere a los microprocesadores.

Los buses de datos que permiten la integración de equipos para la medición y control de variables de proceso reciben la denominación genérica de buses de campo. Un bus de campo es un sistema de transmisión de información (datos) que simplifica enormemente la instalación y operación de máquinas y equipamientos industriales utilizados en procesos de producción.

El objetivo de un bus de campo es sustituir las conexiones punto a punto entre los elementos de campo y el equipo de control a través del tradicional lazo de corriente de 4-20mA o 0 a 10V DC, según corresponda. Generalmente son redes digitales, bidireccionales, multipunto, montadas sobre un bus serie, que conectan dispositivos de campo como PLCs, transductores, actuadores, sensores y equipos de supervisión. Varios grupos han intentado generar e imponer una norma que permita la integración de equipos de distintos proveedores. Sin embargo, hasta la fecha no existe un bus de campo universal.

A continuación, se exponen con detalle los protocolos utilizados para este proyecto.

4.1 MODBUS

4.1.1 Introducción

Modbus es un protocolo de mensajes situado en la capa de aplicación en el modelo OSI, que provee una comunicación Cliente/Servidor entre dispositivos conectados a través de diferentes buses o redes. Este protocolo es el estándar de facto en la industria desde 1979m dónde actualmente habilita millones de conexiones entre dispositivos dedicados a la automatización. La comunidad de internet puede acceder a Modbus a través del puerto 502 reservado en la pila de TCP/IP.

Modbus es un protocolo de Solicitud/Respuesta y ofrece servicios especícos con códigos de función específicos, llamados PDUs (Protocol Data Unit).

Actualmente está implementado usando:

- TCP/IP sobre Ethernet.
- Transmisión serie asíncrona sobre varios tipos de medios (Cable: EIA/TIA-232-E, EIA-422, EIA/TIA-485-A; Fibra, Radio, etc.)
- MODBUS PLUS, un método de paso de token a través de una red local a gran velocidad.

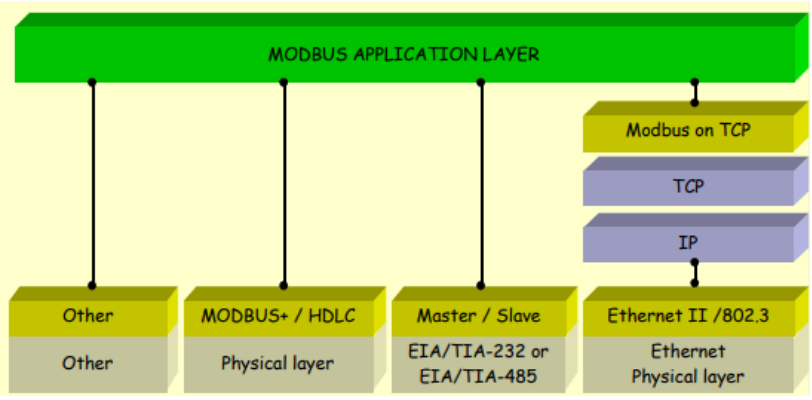


Figura 4-1. Pila de Comunicación Modbus

4.1.2 Descripción de la comunicación

La unidad de datos de la aplicación Modbus está construida por el cliente que inicia una transacción Modbus. La función indica al servidor qué tipo de acción realizar. En la aplicación Modbus, el protocolo establece el formato de una solicitud iniciada por un cliente.

El campo de código de función de una unidad de datos Modbus está codificado en un byte. Los códigos válidos están en el rango de 1 ... 255 decimal (el rango 128 - 255 está reservado y se usa para respuestas de excepción). Cuando se envía un mensaje desde un Cliente a un dispositivo Servidor, el campo de código de función le dice al servidor qué tipo de acción realizar. El código de función "0" no es válido.

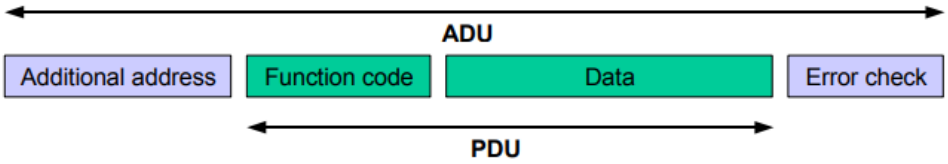


Figura 4-2. Trama general de comunicación Modbus

Los códigos de subfunción se agregan a algunos códigos de función para definir múltiples acciones.

El campo de datos de los mensajes enviados desde un cliente a los dispositivos del servidor contiene información adicional que utiliza el servidor para realizar la acción definida por el código de función. Esto puede incluir artículos como direcciones discretas y de registro, la cantidad de artículos a manejar y el recuento de bytes de datos reales en el campo.

El campo de datos puede ser inexistente (de longitud cero) en ciertos tipos de solicitudes, en este caso es el servidor no requiere ninguna información adicional. El código de función solo especifica el tipo de acción.

Si no se produce ningún error relacionado con la función Modbus solicitada en un ADU (Application Data Units) de Modbus recibido correctamente, el campo de datos de una respuesta de un servidor a un cliente contiene los datos solicitados. Si una se produce un error relacionado con la función Modbus solicitada, el campo contiene un código de excepción que la aplicación del servidor puede usar para determinar la siguiente acción a tomar.

Por ejemplo, un cliente puede leer los estados ON / OFF de un grupo de salidas o entradas discretas o puede leer / escribir el contenido de datos de un grupo de registros.

Cuando el servidor responde al cliente, utiliza el campo de código de función para indicar una respuesta normal (sin errores) o que se produjo algún tipo de error (llamado respuesta excepción). Para una respuesta normal, el servidor simplemente hace eco a la solicitud del Código de función original.

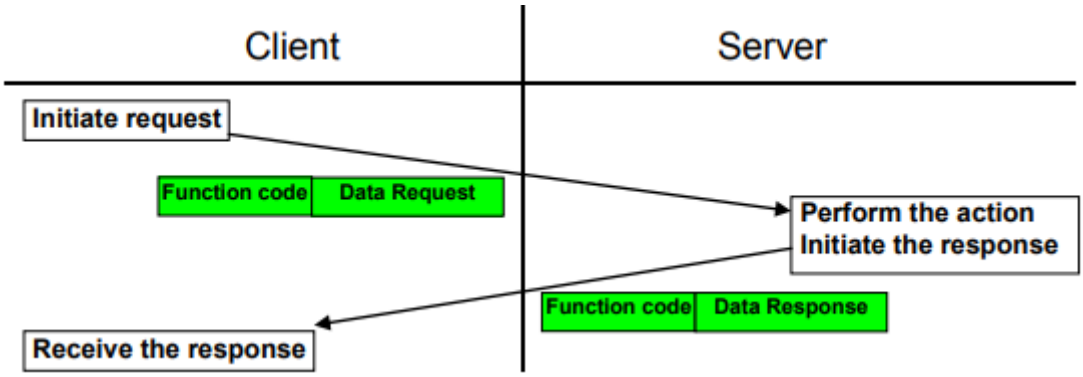


Figura 4-3. Comunicación Modbus Cliente/Servidor sin errores.

El código de operación puede tomar cualquier valor comprendido entre el 0 y el 127 (el bit de más peso se reserva para indicar error). Cada código se corresponde con una determinada operación. Algunos de estos códigos se consideran estándar y son aceptados e interpretados por igual por todos los dispositivos que dicen ser compatibles con Modbus, mientras que otros códigos son implementaciones propias de cada fabricante. Es decir que algunos fabricantes realizan implementaciones propias de estos códigos "no estándar".

				Function Codes			
				code	*Sub code*	*(hex)*	*Section*
Data Access	**Bit access**	Physical Discrete Inputs	Read Discrete Inputs	**02**		**02**	6.2
		Internal Bits Or Physical coils	Read Coils	**01**		**01**	6.1
			Write Single Coil	**05**		**05**	6.5
			Write Multiple Coils	**15**		**0F**	6.11
	16 bits access	Physical Input Registers	Read Input Register	**04**		**04**	6.4
		Internal Registers Or Physical Output Registers	Read Holding Registers	**03**		**03**	6.3
			Write Single Register	**06**		**06**	6.6
			Write Multiple Registers	**16**		**10**	6.12
			Read/Write Multiple Registers	**23**		**17**	6.17
			Mask Write Register	**22**		**16**	6.16
			Read FIFO queue	**24**		**18**	6.18
	File record access		Read File record	**20**		**14**	6.14
			Write File record	**21**		**15**	6.15
Diagnostics			Read Exception status	**07**		**07**	6.7
			Diagnostic	**08**	**00-18,20**	**08**	6.8
			Get Com event counter	**11**		**OB**	6.9
			Get Com Event Log	**12**		**0C**	6.10
			Report Server ID	**17**		**11**	6.13
			Read device Identification	**43**	**14**	**2B**	6.21
Other			Encapsulated Interface Transport	**43**	**13,14**	**2B**	6.19
			CANopen General Reference	**43**	**13**	**2B**	6.20

Figura 4-4. Definiciones de Códigos de Función públicos en Modbus

4.2 OPC UA

OPC Unified Architecture (UA) sigue los requisitos actuales y futuros de las necesidades de comunicación industrial. Unifica y extiende los estándares individuales de la OPC de primera generación (también llamada OPC COM u OPC Classic) utilizando un paradigma de arquitectura orientada a servicios (SOA). Este resultado es una infraestructura de comunicación independiente de la plataforma, escalable y de alto rendimiento. El uso en dispositivos pequeños de tecnología de proceso y medición con sus sistemas operativos especializados es tan posible como el uso en aplicaciones empresariales en máquinas Unix / Linux o Mainframes.

La capa de transporte transforma estos métodos en un protocolo, lo que significa que serializa / deserializa los datos y los transmite a través de la red. Actualmente hay dos protocolos basados en TCP / IP especificados para este propósito. Uno es un protocolo TCP optimizado de alto rendimiento binario y el segundo, un protocolo basado en servicios web. El protocolo binario es obligatorio y es compatible con todas las pilas UA. Los protocolos adicionales son posibles y se pueden agregar cuando sea necesario.

El Modelo de información OPC ya no es solo una jerarquía basada en carpetas, elementos y propiedades, sino

una llamada Red de malla completa basada en nodos. Esta red de nodos también puede transmitir todas las variedades de metainformación y datos de diagnóstico. La imagen más cercana de un nodo sería un objeto, conocido por la programación orientada a objetos (OOP). Puede estar compuesto de variables que se pueden leer o escribir, métodos que se pueden llamar y eventos que se pueden disparar. Un evento contiene, entre otras cosas, un tiempo de notificación, un mensaje y una gravedad. Los nodos se utilizan para procesar datos, así como para todos los demás tipos de metadatos. El espacio de nombres OPC recientemente modelado ahora contiene el modelo de tipo utilizado para describir todos los tipos de datos posibles también.

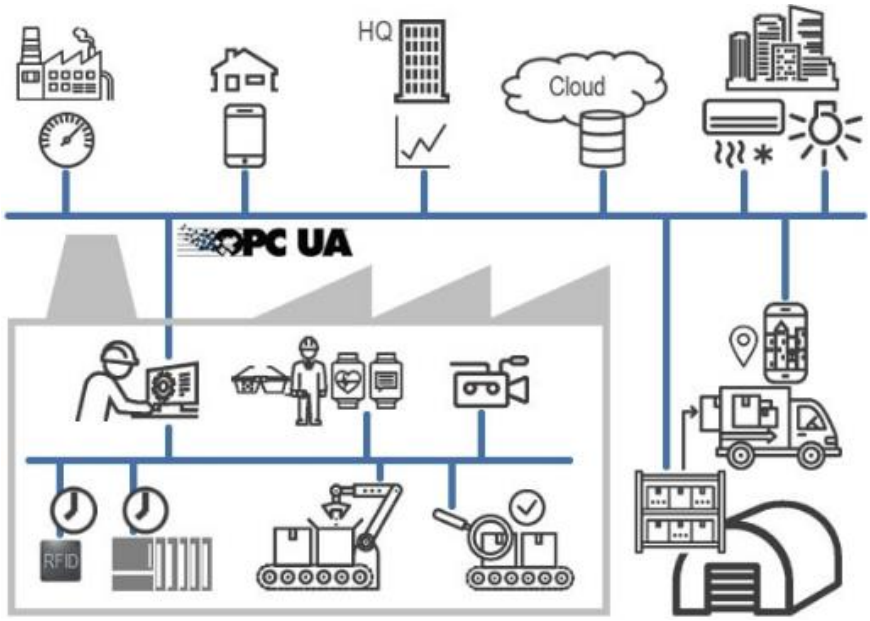


Figura 4-5. Mapa de integración de OPC UA con diferentes dispositivos.

Sobre la base de esta nueva arquitectura, otras organizaciones están especificando su propia fuente de información, las llamadas especificaciones complementarias. Una de las primeras de estas especificaciones es el modelo DI (integración de dispositivos). Describe dispositivos y forma una base para otras especificaciones complementarias como ADI, PLCopen o FDI, que por su parte definen sus propios modelos de información. El software del cliente tiene la capacidad de verificar cuál de los llamados Perfiles admite un servidor. De esta forma, los clientes pueden detectar si un servidor solo admite la funcionalidad DA (acceso a datos) o, además, A&C (alarmas y condiciones), HDA (acceso a datos históricos), etc.

Las características adicionales importantes de OPC UA son:

- Arquitectura orientada a servicios utilizando un paradigma asíncrono de solicitud / respuesta.
- Combina todas las características de las especificaciones anteriores "OPC clásico".
- Operaciones masivas para todas las operaciones de acceso.
- Transmisión amigable con el ancho de banda.
- Soporte de redundancia en el lado del cliente y servidor, redundancia múltiple.
- Latido para monitorizar la conexión en ambas direcciones; es decir, tanto el servidor como el cliente reconocen fallas.
- Almacenamiento en búfer de datos y acuses de recibo de datos transmitidos.
- Las conexiones perdidas ya no conducen a la pérdida de datos; los paquetes de datos perdidos pueden recuperarse nuevamente.

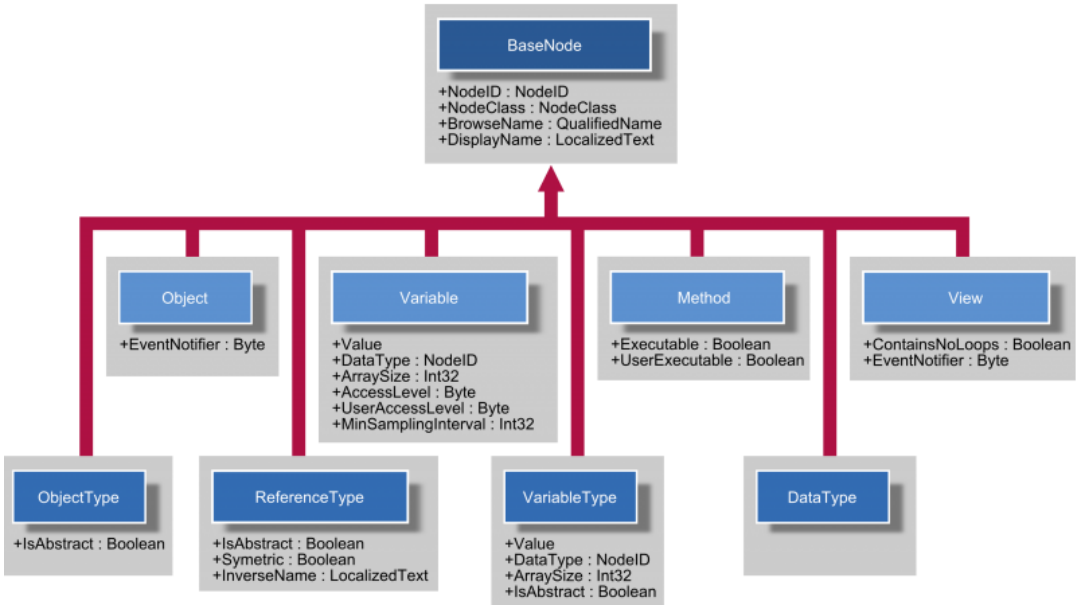


Figura 4-6. Modelos de datos de OPC UA

OPC UA define un modelo de objeto genérico que incluye el sistema de tipos asociado. Además de este modelo de datos, se han definido reglas para describir cómo transformar cada sistema físico en un modelo conforme con OPC UA para representarlo en un servidor OPC UA. Se puede describir todo tipo de dispositivo, función e información del sistema utilizando este metamodelo. El sistema de tipo base admite relaciones entre objetos, las denominadas referencias, así como la herencia múltiple. Por lo tanto, se puede comparar con un lenguaje de programación moderno orientado a objetos. El modelo base proporciona tipos de objetos y variables, así como tipos de referencia y datos. Basado en este modelo, OPC UA puede representar todo tipo de datos, incluidos sus metadatos y semántica.

5 EJECUCIÓN

En este apartado se abordará la parte práctica del trabajo, tratando la configuración de los dispositivos, la comunicación entre ellos y cómo se han añadido las funcionalidades al sistema.

No será objeto de este apartado la instalación del software que se utiliza de apoyo para conseguir el resultado final, ni la instalación y configuración del servidor virtual en base Linux.

5.1 Puesta en marcha del Smartlink

Como tarea inicial se aborda la configuración del Smartlink con los Powertags. En este proyecto se tienen de partida todos los dispositivos conectados en el cuadro de baja tensión, entre ellos y a la red de internet local. Gracias al servidor DHCP del Smartlink, autómaticamente al conectar el dispositivo a la red, esta asignará una dirección ip dentro del rango del servidor DHCP establecido por el servidor.

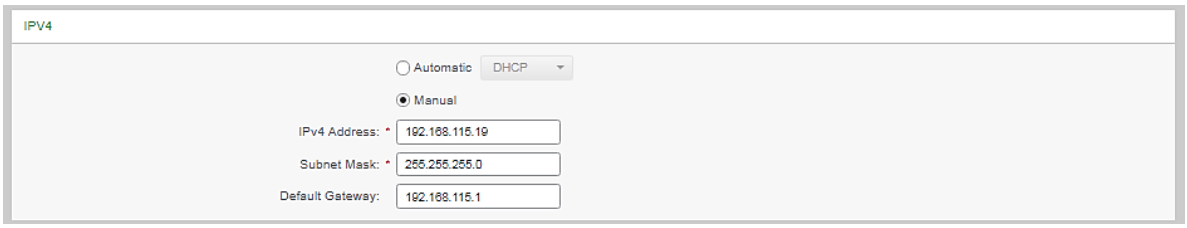


Figura 5-1. Configuración manual del protocolo IPv4 del Smartlink.

Apuntando a la dirección ip del Smartlink en el buscador de internet, se nos redirige al portal de configuración de este, donde primero se deben introducir las credenciales de usuario.

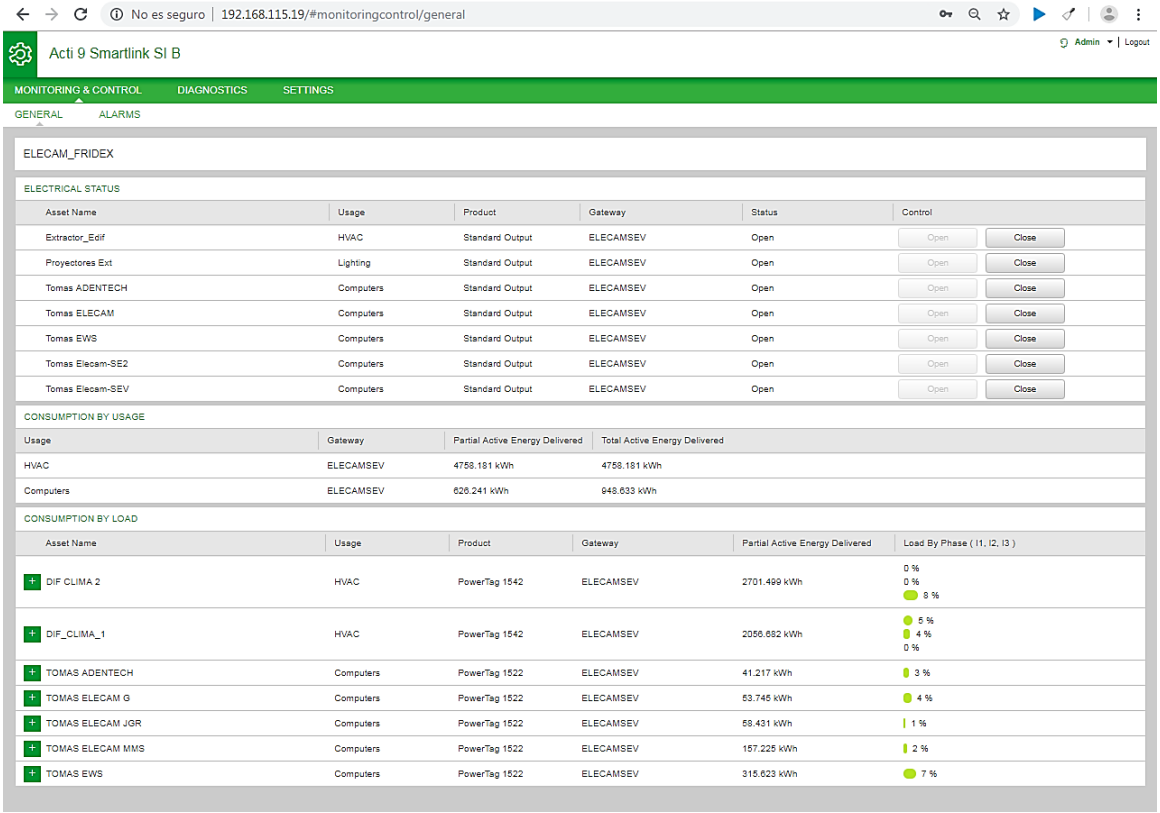


Figura 5-2. Panel de monitorización de Smartlink SI B

Una vez dento del portal web lo primero que se realizará será cambiar la configuración DHCP del dispositivo por una configuración de red con ip fija, ya que en caso contrario el servidor podría dar una dirección distinta al dispositivo y esto derivaría en un error de comunicación con el autómata, que como ya se verá en el siguiente apartado, está programado para conectarse al Smartlink a través de una dirección ip específica. Tras ello, accedemos a la configuración de dispositivos, donde configuraremos las entradas y salidas digitales y los dispositivos Modbus wireless Powertags.

- Entradas y Salidas Digitales: Gracias a los planos eléctricos proporcionados por el diseñador eléctrico del cuadro, damos nombre a las diferentes entradas y salidas digitales para una mejor monitorización y control posterior por parte del autómata.
- Powertags: En este caso se utiliza un método sencillo de configuración gracias a la posibilidad de visualizar todos los dispositivos Powertags activos con la opción de descubrir dispositivos wireless. Una vez se hayan descubierto todos los powertags activos, se realiza una inspección visual para averiguar qué dispositivo descubierto pertenece a línea a medir correspondiente, una vez realizado este proceso, se les da un nombre correspondiente y un número de esclavo para identificarlos.

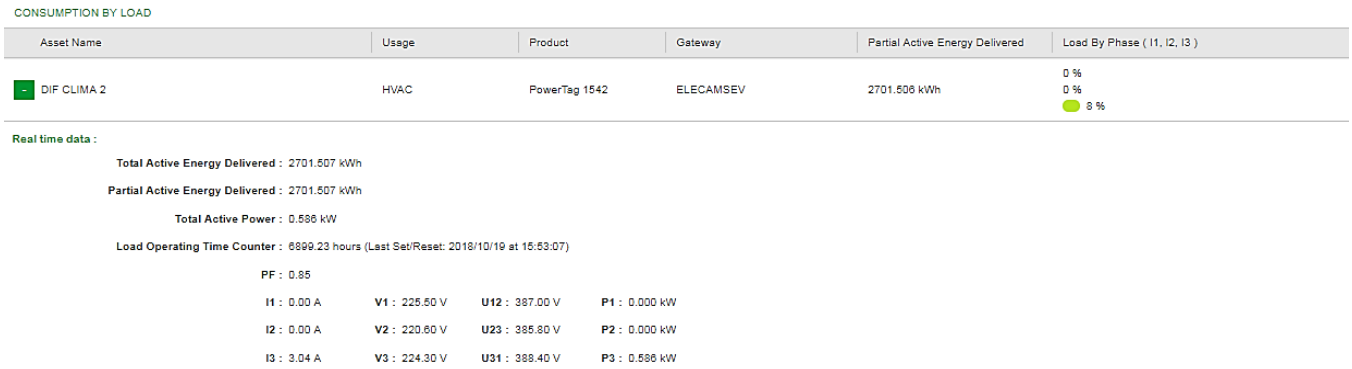


Figura 5-3. Detalles de las medidas de un sensor Powertag

Una vez realizada la puesta en marcha del Smartlink, en la página principal del portal web se pueden visualizar y las entradas y salidas digitales y visualizar los valores medidos por los powertags. También se pueden configurar alarmas que quedarán notificadas en el portal, como sobrecargas. Estas funcionalidades del Smartlink son realmente útiles, pero estos datos son en su mayoría instantáneos, salvando la energía consumida por línea, y las alarmas configuradas.

5.2 Configuración del PLC

El objetivo principal del control a través del autómata será añadir funcionalidades que no son posibles con el sistema de Smartlink con Powertags por sí solo.

Antes de comenzar la programación del autómata se procede a conectarlo a la red local, para que el autómata pueda ser visible por el software SoMachine. En este caso en concreto se decide realizar de esta forma ya que el cuadro está situado en un lugar separado de la zona de trabajo, por lo que se decide operar a través de la red local en vez de por cable USB conectado diréctamente al dispositivo.

El autómata y el Smartlink están conectados a una tarjeta Switch de 4 puertos Ethernet de expansión del PLC, la cual se sitúa a su izquierda y conectamos los cables UTP con conectores RJ45, de cada dispositivo al Switch

y del Servidor al Switch, de esta forma están conectados todos los dispositivos a la red local.

Una vez esté conectado el autómata a la red, cuando accedemos al software SoMachine, si está todo correcto, aparecerá el dispositivo, sus características y el programa que está cargado en él. Lo primero será configurar los parámetros de red de forma que el autómata disponga de una ip fija, al igual que el Smartlink, así evitaremos conflictos con los distintos dispositivos conectados a la misma red local, como PCs, impresoras, etc.

5.3 Comunicación a través de Modbus TCP

La lectura y escritura de variables entre el autómata y el Smartlink se realiza a través del protocolo Modbus TCP/IP. Esta comunicación se puede realizar a través de una funcionalidad de SoMachine llamada IO Scanner, en la cual se introduce la dirección ip del maestro Modbus, el número de esclavo, y las palabras de memoria que se quieren leer. Se decide no utilizar esta funcionalidad del sistema, ya que abre un socket por cada elemento que se lee o escribe, lo cual no es eficiente en el sistema que se va a controlar ya que el autómata solo puede tener 16 Sockets TCP abiertos a la vez.

5.3.1 ADDM

Se dispone de un bloque ADDM donde se introduce la dirección del puerto ethernet del autómata, la dirección ip del maestro Modbus, el puerto TCP que utiliza (por defecto en el protocolo es el 502) y el número de esclavo (En el caso de la lectura de parámetros del maestro modbus, habría que eliminar el número de esclavo). Para realizar el muestreo se utiliza un bloque de función BLINK a la que se le introduce como entradas los tiempos de on/off de la salida del bloque, que se utiliza como entrada del enable del bloque ADDM, así se abrirá y cerrará el socket, en caso de tener muchos más dispositivos se podría hacer una secuencia de apertura y cierre de sockets para que nunca se supere el máximo de 16. La salida del bloque ADDM será la dirección en formato ADDRESS que se le pasará a los bloques de lectura y escritura de variables. La salida DONE, que supone que el socket se ha abierto correctamente, se utiliza como entrada para ejecutar las lecturas y escrituras en los bloques.

5.3.2 READ_VAR y WRITE_VAR

Para leer y escribir variables se utilizan los bloques de funciones READ_VAR Y WRITE_VAR respectivamente. En ambos casos como entradas se les pasarán los espacios de memoria Modbus definidos en el datasheet del Smartlink SI B, teniendo en cuenta que en el protocolo Modbus direcciona la memoria comenzando desde el número "1" y en la memoria del dispositivo se guarda comenzando desde el número "0", por tanto al leer y escribir desde los bloques de funciones se les pasará el espacio de memoria o la entrada de registro que venga reflejada en la tabla de direccionamiento Modbus restando una unidad. Se le pasa la dirección que procede del bloque ADDM, los espacios de memoria que se van a leer o escribir y un puntero al vector donde se van a guardar que se van a enviar. Por último, se debe especificar el tipo de objeto que se va a leer o escribir en la entrada ObjectType, en la cual se introduce un número constante entero sin signo que no es otro que el código de función Modbus (Figura 4-4). En SoMachine, utilizando los bloques de funciones READ_VAR y WRITE_VAR solo podremos utilizar cuatro códigos de función que vienen definidos en el tipo estructura ObjectType: I(Read Coil), IW(Read Input Register), Q(Write Single Coil), MW(Write Multiple Register). La única salida que vamos a utilizar será la salida DONE que marcará que se ha realizado con éxito la lectura o escritura, por tanto se pasa a la entrada 'Execute' del siguiente bloque READ_VAR o WRITE_VAR.

5.3.3 Tratamiento de variables en memoria

Una vez estamos leyendo las variables a través de Modbus TCP y las tenemos guardadas en vectores, es necesario pasar esas variables al formato que viene especificado en las tablas Modbus del dispositivo, en este caso todos los parámetros están en formato flotante de 32-bits, lo cual quiere decir que tendremos un vector de dos componentes de 16-bits cada uno, o lo que es lo mismo, dos "palalabras".

Para pasarlo a formato flotante, habrá que mover ese vector a un espacio de memoria libre, de forma que las componentes del vector sean consecutivas, siendo la primera de ellas en un espacio de memoria par y la segunda en el espacio consecutivo. En SoMachine se especifican las palabras de memoria como "%MWx" donde x es el espacio que ocupan, por tanto, tras leer o escribir las variables hay que guardarlas en memoria como se especifica anteriormente, y tras ello, tendremos automáticamente una variable "doble" (32-bits) en el espacio de memoria %MDx/2, tras esto, definimos una variable global en ese mismo espacio de memoria para poder tener control sobre ella, y la definimos como REAL, lo cual en SoMachine significa que será de formato flotante.

En el caso de las entradas y salidas digitales, leemos la palabra de memoria donde se alojan los bits asociados a las entradas y se desglosa dicha palabra con el bloque WORD_AS_BIT, por tanto, podremos asociar un bit a cada entrada. Con las salidas hacemos el mismo trabajo, pero a la inversa, creamos una palabra que será la que se introducirá como orden de cerra o abrir los contactos, siendo dicha palabra construida a partir del bloque BIT_AS_WORD, donde se le pasan los bits asociados a la salida que se quiera cambiar y se forma la palabra que se introducirá en el bloque WRITE_VAR.

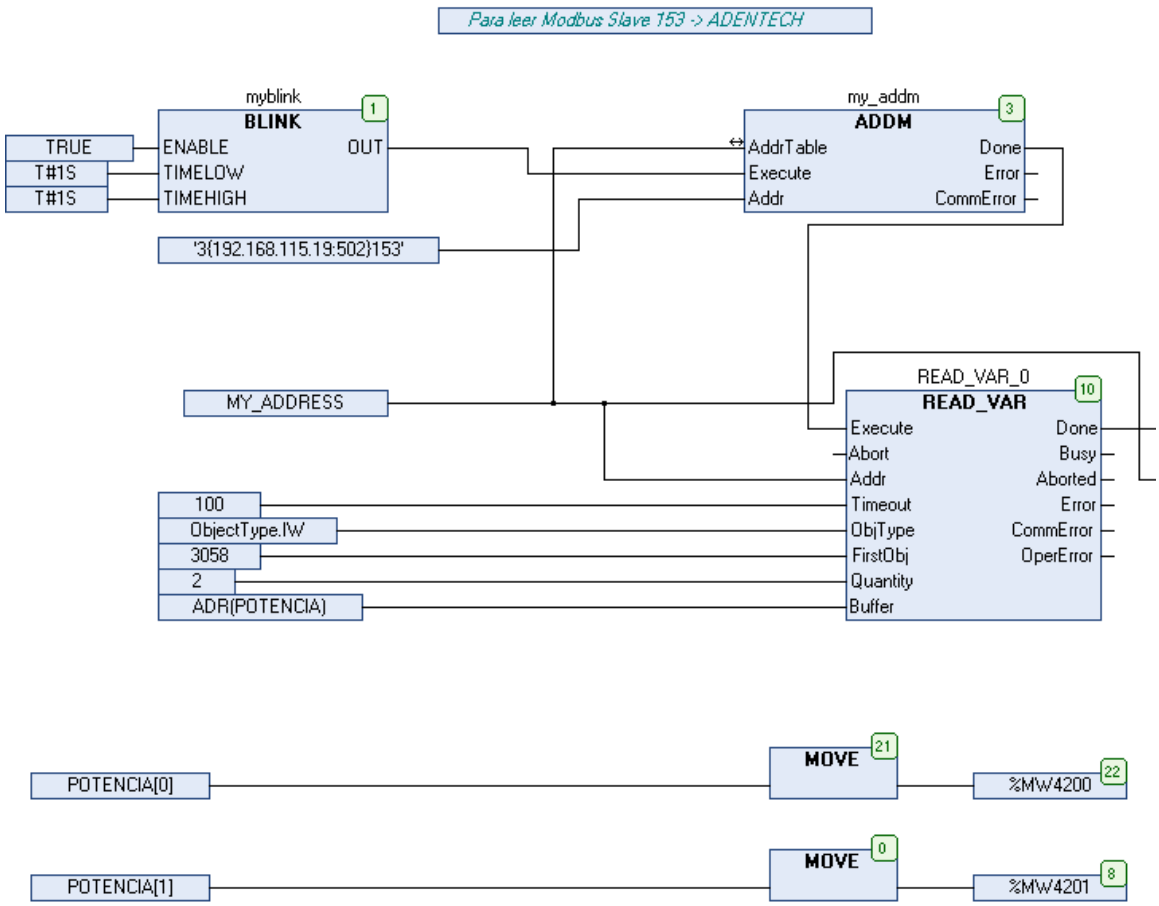


Figura 5-4. Comunicación Modbus TCP y reserva en memoria para una variable tipo Float32

5.4 Funciones del Sistema

Tras haber realizado la programación mediante la cual se recogen todos los datos en tiempo real, se programan las funcionalidades particulares del sistema, la mayoría de ellos programados en lenguaje CFC, con la ayuda de las librerías de Codesys y los bloques de funciones que lo forman.

5.4.1 Control Horario

Se realiza una programación horaria de las líneas que están conectadas a las señales digitales del Smartlink, la cual se realiza de forma que se puedan cerrar los disyuntores de línea a una hora y un minuto determinado y abrir a otra. La particularidad es que, al producirse la apertura de las líneas, tras acabar la jornada laboral, se programa un reconocimiento de estado de apertura, de este modo nunca se quedará encendida ninguna línea en caso de que el sistema falle si se programase con una simple señal de apertura en un momento determinado. Se realiza el control mediante bloques biestables RS, en los cuales prevalece el reset sobre el set.

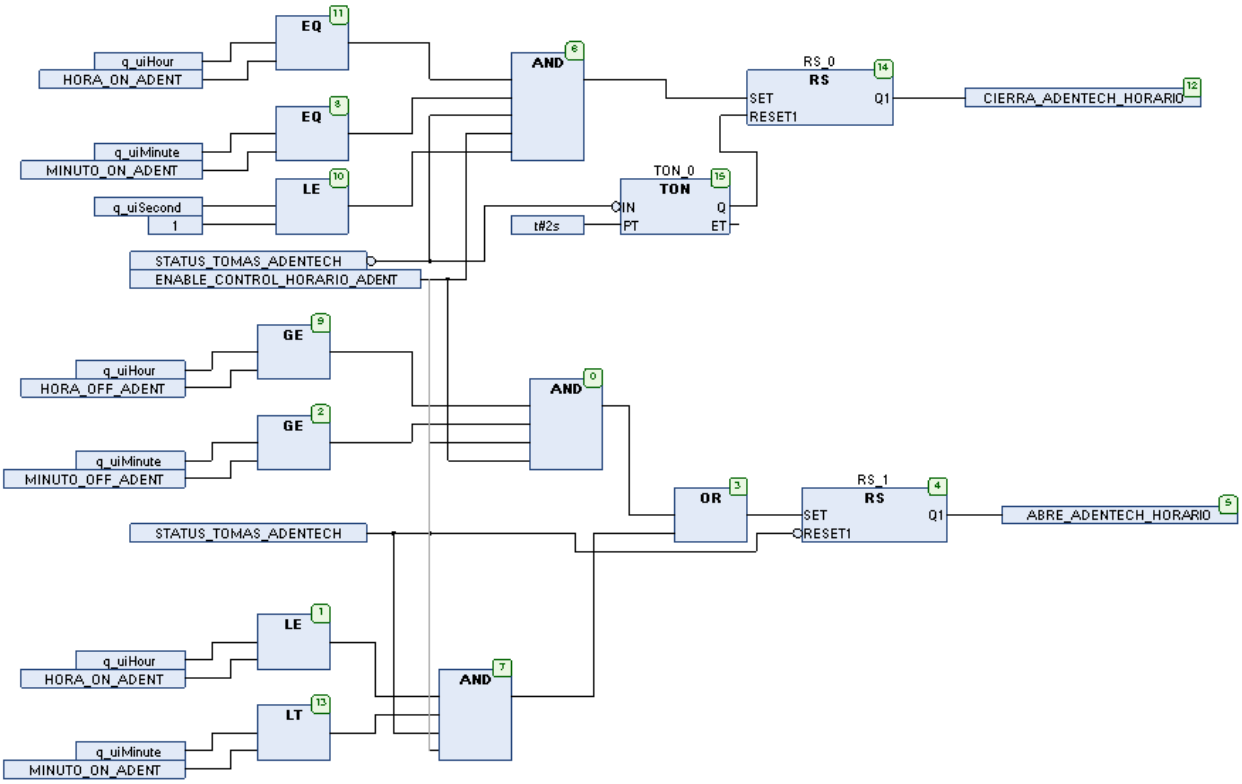


Figura 5-5. Control horario de las tomas de la nave del Smartlink

5.4.2 Energía Consumida

Los sensores Powertag poseen un contador de energía parcial y total consumida, pero no es realmente útil en este caso, ya que tras revisar la factura eléctrica de la empresa se aprecia que está discriminada en dos tramos: valle, de 12 de la noche a 12 de la mañana, y punta, de 12 de la mañana a 12 de la noche. Para contabilizarlo, creamos una variable de energía por línea medida en kWh (kilovatios hora) y la aumentamos una vez por segundo sumando esa misma variable a la potencia actual de la línea, dividido entre 3600, que son los segundos que hay en una hora. Puesto que el autómata da la posibilidad de crear hasta 12 tareas distintas, creamos una

separada de la tarea principal, con un tiempo de ejecución de 1 segundo, por lo que no deberemos el tiempo de muestreo del contador. Tras ello, se discrimina en los dos tramos, haciendo que solo esté activo en su horario.

En caso de la energía diaria se reiniciará al llegar a las cero horas y cero minutos de cada día, en el caso de la energía semanal se reiniciará al llegar a las cero horas y cero minutos del lunes y en caso de la energía mensual, se reiniciará a las cero horas y cero minutos del día 1 de cada mes.

```
1   IF Q_UIDAY = 1 AND Q_UIHOUR = 0 AND Q_UIMINUTE = 0 AND Q_UISECOND = 0 THEN
2   CONSUMO_150_MES_P       := 0;
3   CONSUMO_151_MES_P       := 0;
4   CONSUMO_152_MES_P       := 0;
5   CONSUMO_153_MES_P       := 0;
6   CONSUMO_154_MES_P       := 0;
7   CONSUMO_155_MES_P       := 0;
8   CONSUMO_156_MES_P       := 0;
9   CONSUMO_150_MES_V       := 0;
10  CONSUMO_151_MES_V       := 0;
11  CONSUMO_152_MES_V       := 0;
12  CONSUMO_153_MES_V       := 0;
13  CONSUMO_154_MES_V       := 0;
14  CONSUMO_155_MES_V       := 0;
15  CONSUMO_156_MES_V       := 0;
16
17  END_IF
18      IF Q_UIHOUR >= 12 AND Q_UIHOUR < 22 THEN
19
20          CONSUMO_150_MES_P := CONSUMO_150_MES_P + PKW_150/3600;
21          CONSUMO_151_MES_P := CONSUMO_151_MES_P + PKW_151/3600;
22          CONSUMO_152_MES_P := CONSUMO_152_MES_P + PKW_152/3600;
23          CONSUMO_153_MES_P := CONSUMO_153_MES_P + PKW_153/3600;
24          CONSUMO_154_MES_P := CONSUMO_154_MES_P + PKW_154/3600;
25          CONSUMO_155_MES_P := CONSUMO_155_MES_P + PKW_155/3600;
26          CONSUMO_156_MES_P := CONSUMO_156_MES_P + PKW_156/3600;
27
28      END_IF
29
30      IF Q_UIHOUR >=22  OR Q_UIHOUR < 12 THEN
31
32          CONSUMO_150_MES_V := CONSUMO_150_MES_V + PKW_150/3600;
33          CONSUMO_151_MES_V := CONSUMO_151_MES_V + PKW_151/3600;
34          CONSUMO_152_MES_V := CONSUMO_152_MES_V + PKW_152/3600;
35          CONSUMO_153_MES_V := CONSUMO_153_MES_V + PKW_153/3600;
36          CONSUMO_154_MES_V := CONSUMO_154_MES_V + PKW_154/3600;
37          CONSUMO_155_MES_V := CONSUMO_155_MES_V + PKW_155/3600;
38          CONSUMO_156_MES_V := CONSUMO_156_MES_V + PKW_156/3600;
39
40      END_IF
```

Figura 5-6. Discriminación de los consumos mensuales en lenguaje ST.

5.4.3 Estadísticas básicas

Se calculan a partir de los bloques de función la máxima, mínima y la media de la potencia actual de cada línea, reiniciándose cada día. Se programa de una forma muy sencilla, solo introducimos al bloque la entrada que será la potencia actual y el reset se dispara cuando cada día se llega a las cero horas y cero minutos.

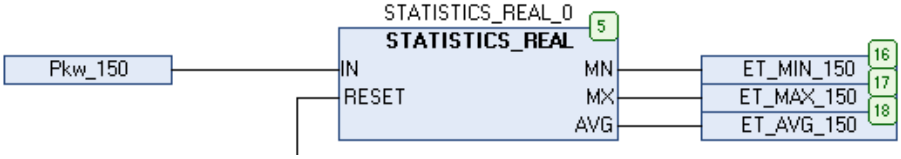
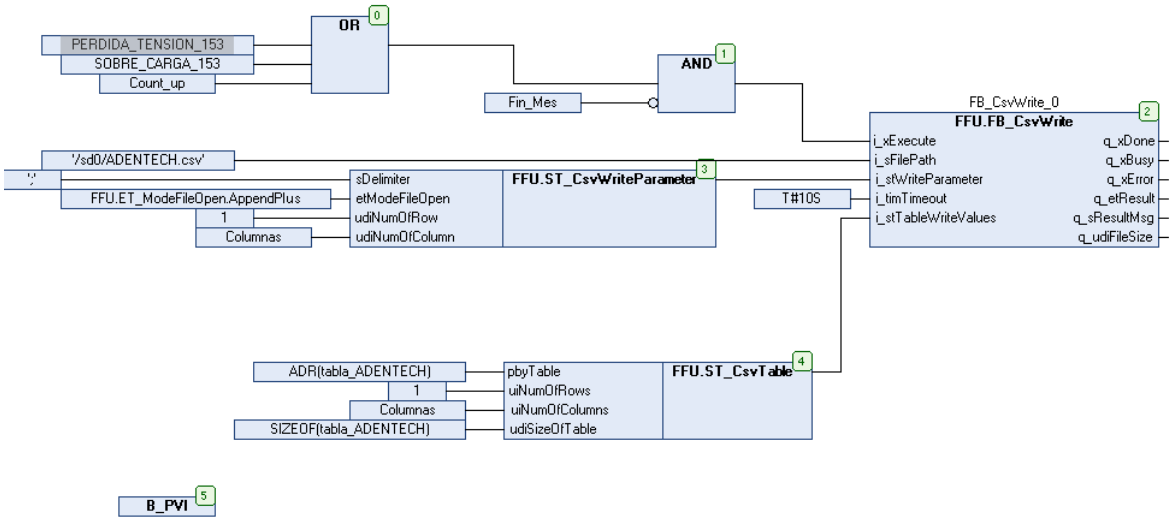


Figura 5-7. Estadísticas sobre la potencia actual por línea.

5.4.4 Generación de archivos '.csv'

Quizás la funcionalidad más importante del sistema, para cada sensor Powertag se crea un vector con todos los valores recogidos y los calculados que vayamos a guardar en un archivo, y cada 5 minutos escribimos en una tabla en formato csv, la cual se puede abrir con el software Microsoft Excel, el vector con los valores, con la particularidad que dicho vector cada vez que se escribe, lo hace de forma consecutiva a la vez anterior, creándose un registro en el archivo. Cada mes, y por motivos de límite de almacenamiento, se sobreescribirán los archivos con una tabla vacía creada, de forma que nunca se llegue a sobrepasar el límite de la memoria sd. Por supuesto, antes de sobreescribir esos archivos se descargarán gracias al servidor FTP alojado en el autómata. En siguientes apartados se abordará una forma alternativa de guardar el registro de datos, en este caso desde un servidor externo

Figura 5-8. Escritura de archivo csv programado en lenguaje CFC.



5.4.5 Servidor OPC UA

El autómata contiene la funcionalidad de servidor OPC UA integrada, lo que se utilizará para comunicar con distintas aplicaciones externas. En el caso de SoMachine, entramos en la pestaña del controlador llamada 'Servidor OPC UA' y activamos el servidor, sabiendo que una vez activo, las variables que puedan ser leídas por los clientes tienen dos componentes, un índice definido por un número entero y un identificador que en nuestro caso es una cadena de caracteres, o lo que es lo mismo, el nombre de las variables que hemos definido en SoMachine. Para realizar la comunicación con el servidor, deberemos abrir un socket de escucha tcp a la dirección: 'opc.tcp:// ' + 'url del autómata' + ':' + 'puerto'. En este caso concreto, al haberse programado un grupo de usuarios, deberemos presentar credenciales al acceder a las variables del servidor desde los clientes.

Por último, creamos un obeto en la aplicación de SoMachine que permite seleccionar las variables disponibles en el servidor hasta un máximo de 3000.

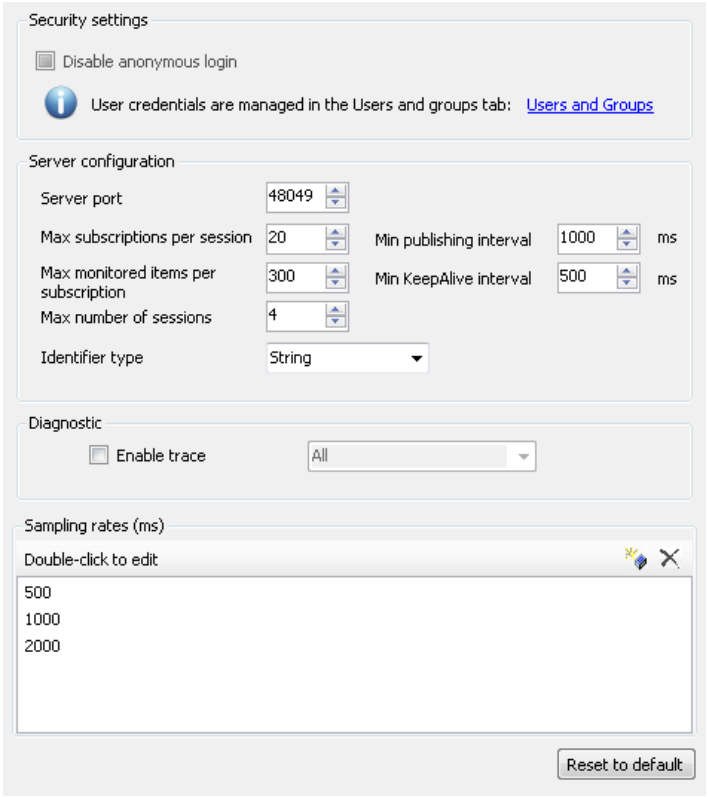


Figura 5-9. Configuración del Servidor OPC UA en el autómata.

5.4.6 SCADA en Web Server

A través del web server, apuntando a la dirección ip del autómata más el puerto 8080, y el nombre de la aplicación de la visualización, se puede visualizar y controlar todo el sistema a modo de Scada. Se programa a través de la aplicación de diseño de páginas visuales en SoMachine, en la que tenemos figuras y herramientas muy básicas pero programables.

Se realiza el diseño de la aplicación Scada programando la aplicación para que sea sencilla e intuitiva de utilizar. Primero se diseña una página principal en donde se sitúa la planta de la nave y sus líneas dispuestas en su situación real. Por cada elemento se dispone una caja donde se puede apreciar un botón de apagado y encendido que conmuta de imagen dependiendo de su estado, y un botón de control horario en en el caso de las salidas digitales que no llevan Powertag asociado. El control horario abre una ventana pop-up que pregunta las horas de on/off y habilita el control con un botón de 'enable'. A las páginas asociadas a los sensores Powertag se acceden 'clickando' en las mesas, en caso de las líneas asociadas a las mesas de trabajo, las cuales tienen un tratamiento visual que permite el reconocimiento de que realmente son un objeto con el que se puede interactuar. En el caso de las líneas de clima, se accede a los parámetros eléctricos a través de un botón situado en la propia caja.

Al acceder a los parámetros eléctricos de las líneas en tiempo real, se pueden observar todos los valores leídos en tiempo real dentro de un cuadro de texto y una gráfica temporal, donde se observa una traza de la potencia consumida por la línea en cuestión en las últimas 24 horas.

Figura 5-10. Página principal de la visualización web del autómata.

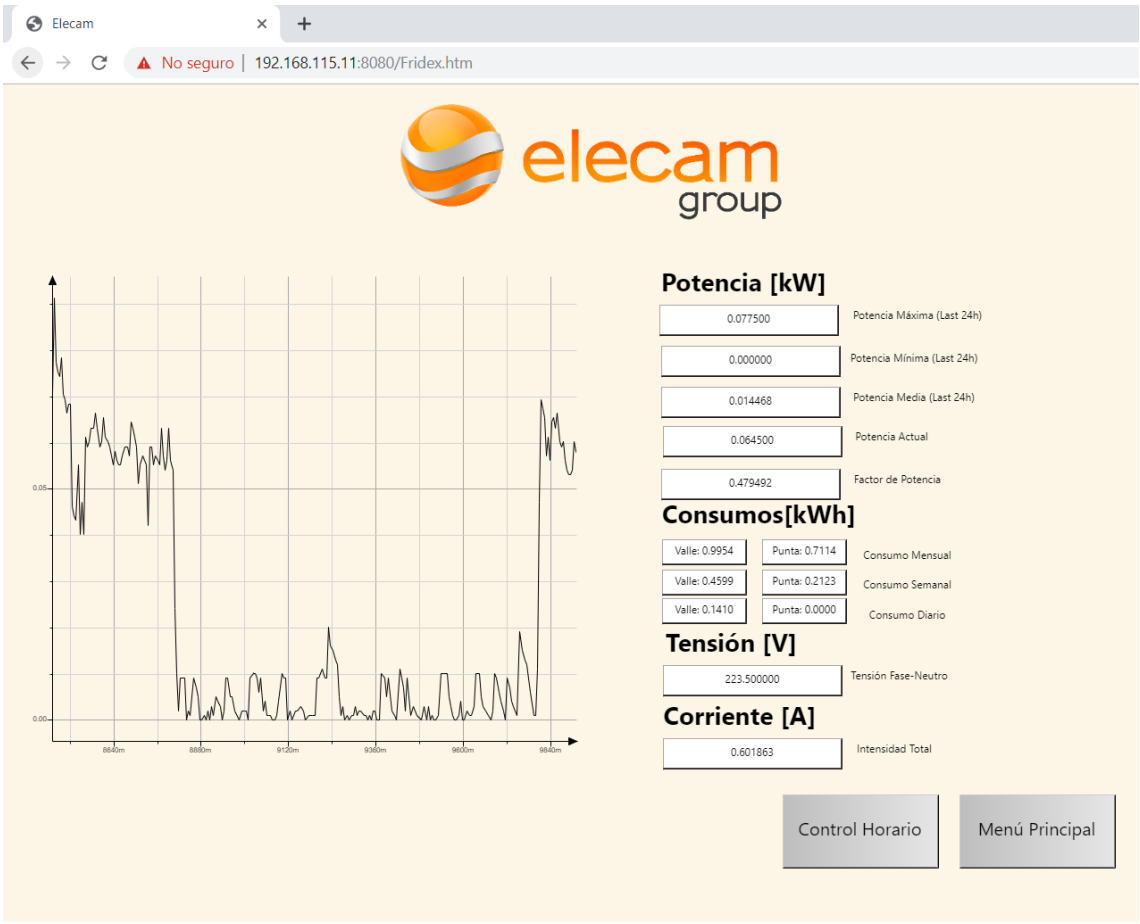


Figura 5-11. Página de visualización de parámetros eléctricos en web server.

5.5 Realidad Aumentada

Una funcionalidad adicional que acerca el trabajo aún más a las nuevas tecnologías IoT es una integración del sistema con una aplicación de realidad aumentada y un sistema de almacenamiento en una base de datos MySQL.
En este caso lo que se va a hacer es simular un servidor en base Linux, con una imagen virtual de Ubuntu 16.04 alojado en nuestro portátil, de esta forma podremos utilizar todas las aplicaciones y probarlas en caso de adquirir el servidor industrial. La instalación y puesta en marcha del servidor y sus aplicaciones se desarrollará en el apartado Anexo.

Primero habrá que descargarse de la página oficial de Schneider Electric el software Augmented Operator Advisor, en este caso lo haremos mediante la versión de prueba. Una vez se haya descargado lo único que debemos hacer es iniciar las aplicaciones teniendo especial cuidado en que los puertos de cada aplicación que utiliza el software estén disponibles para su uso.

Una vez hecho esto se procede a entrar a la aplicación de desarrollo de proyectos de realidad aumentada, la cual se ejecuta en línea.

Esta aplicación es simple e intuitiva, lo primero que se hará será crear las escenas que se van a reconocer, en nuestro caso el cuadro desde el exterior y cada una de las líneas.

Tras ello introducimos las variables que se van a visualizar, con el mismo nombre que en el autómata para evitar confusiones. Finalmente modificamos la escena, añadiendo los valores de las variables situadas sobre los elementos que están midiendo, añadimos también el logo de la empresa para que sea visible desde el exterior, y en algunos casos añadimos la url de los pdf con las especificaciones técnicas de ciertos elementos como el autómata o el analizador de redes. También se pueden introducir pasos a seguir en caso de avería, pero en este caso concreto no se va a programar.

Las variables las situamos en formato de texto introduciendo '{0}', lo cual hará que al ´cargar´ la variable en el servidor http del que se sirve la aplicación, se podrá visualizar.

Tras terminar el diseño del proyecto, deberemos cargarlo en la aplicación, para ello hacemos build y lo descargamos, y descomprimiremos en el directorio de proyecto de Nginx.

Para 'mandar' las variables desde el autómata a la aplicación de realidad aumentada vamos a utilizar node-red. Node-red está disponible en el software Augmented Operator Advisor, pero en nuestro caso lo instalaremos independiente del software en el servidor virtual. Lo que se hará será descargar la librería específica de Augmented Operator Advisor que contiene los dos módulos que nos hacen falta para realizar la tarea, además instalamos también el paquete de módulos de OPC UA, que nos servirá para comunicarnos con el autómata.

Tras esto, programamos los dos módulos necesarios para recoger las variables del autómata a través de OPC UA:

- Inject: Especificamos el tiempo de muestreo y las direcciones de las variables
- Read: Leemos las variables del servidor OPC UA, especificando la dirección del autómata y el puerto que utiliza. Como salida, este módulo dará como resultado un vector de msg.payload, que contiene los valores de las variables leídas.

Para que la aplicación de realidad aumentada reciba las variables leídas, creamos un par de módulos por cada variable.

- Augmented-operator-variable: Este módulo recibe la salida del módulo Read OPC UA. Especificamos el tipo y nombre de la variable, y especificamos la ruta en la que se sitúa el valor de dicha variable en la entrada, que será msg.payload[k].value, donde k puede valer desde 0 hasta el número de variables que se están leyendo menos uno.
- Augmented-operator-server: Este módulo recibe la salida del módulo anterior, y simplemente se especifica el puerto que recibe o manda la información (El puerto que se especifica como Python en el software, en nuestro caso 8082) y el método: Put para mandar la información al servidor, y Get para recibirla, este último no lo utilizaremos.

Por último, descargamos la aplicación en nuestro dispositivo móvil, y especificamos la dirección y el puerto del servidor http Nginx, e introduciendo nuestras credenciales, que habremos creado previamente en el Augmented Operator Advisor Manager, entramos en la aplicación.

Se puede observar que se nos abre automáticamente la cámara, y sin realizar ningún tipo de acción, al enfocar a los objetos que hemos programado como áreas, aparecerán los objetos en realidad aumentada.

Figura 5-12. Runtime de la app Augmented Operator Advisor en dispositivo Android

5.5.1 Integración de MySQL con el proyecto

MySQL es un sistema de gestión de bases de datos relacional desarrollado bajo licencia dual: Licencia pública general/Licencia comercial por Oracle Corporation y está considerada como la base datos de código abierto más popular del mundo,12 y una de las más populares en general junto a Oracle y Microsoft SQL Server, sobre todo para entornos de desarrollo web.

Está desarrollado en su mayor parte en ANSI C y C++.4 Tradicionalmente se considera uno de los cuatro componentes de la pila de desarrollo LAMP y WAMP.

Para los propósitos de nuestro proyecto, instalaremos el paquete mysql en el servidor Ubuntu virtual, y gestionaremos la información de la base de datos desde Node-Red.

El por qué de una base de datos SQL es la posible ampliación del proyecto pasando del nivel local a la nube, lo cual sería la evolución lógica de este trabajo. En un servidor Cloud tendríamos la misma estructura de trabajo, en vez de crear unos archivos en un sistema de almacenamiento físico, podríamos contratar una gran cantidad de espacio y almacenar toda esa información directamente en bases de datos relacionales SQL de manera que en en el futuro se pueda integrar con sistemas empleados para el estudio de dichos datos.

Para la puesta en marcha de la base de datos, una vez hayamos instalado los paquetes y abierto el

puerto 3306, por donde se comunicará la base de datos con el exterior, y añadido un usuario con acceso a todas las bases de datos, crearemos una base de datos para cada línea, y crearemos una columna para cada dato.

En Node-Red, añadiendo al flujo que ya tenemos creado para la aplicación de realidad aumentada el módulo "mysql" y un módulo de función:

- Función: Pasamos al módulo como entrada la salida del módulo Read OPC UA, y programamos el mensaje de salida de la siguiente forma:
    - Topic: Será una cadena de caracteres donde se especificará el 'Query' que entrará a la base de datos, en este caso tendrá esta forma:
      "INSERT INTO *table_name* (*column1*, *column2*, *column3*, ...)
      VALUES (*?*,*?*,*?* ...);"
    - Payload: En este caso será el valor a escribir, que sustituirá en el topic a "?", por tanto igualaremos al valor que entra en el bloque de esta forma: "msg.payload = [msg.payload[k].value,msg.payload[j].value,…]", donde k y j son números enteros desde 0 hasta el número máximo de variables leídas menos uno.
- Mysql: Se le pasa como entrada la salida del módulo de función y dentro de este módulo se especifica el nombre de la base de datos, la dirección ip donde se sitúa, el puerto mediante el cual se comunica, y las credenciales de usuario con privilegios para acceder a dicha base de datos.

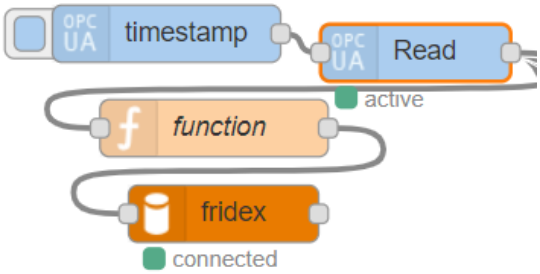
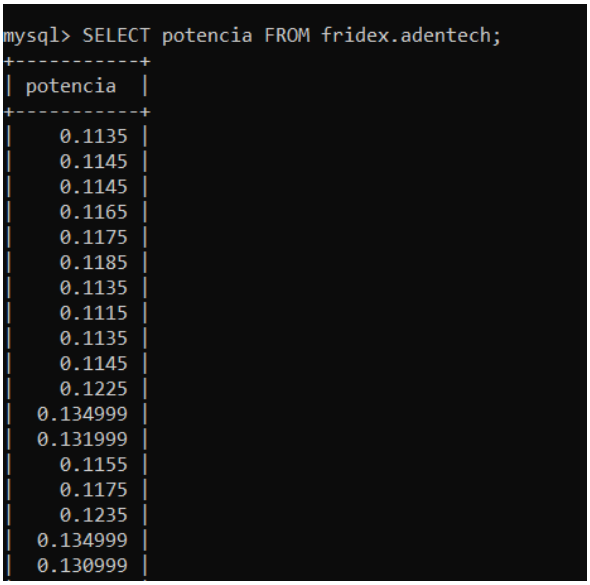


Figura 5-13. Integración en Node-Red de OPC UA con MySQL

Para visualizar los valores que se están escribiendo en la base de datos, entramos en el terminal del servidor virtual, y entramos a la aplicación MySQL con el comando 'sudo mysql -u root -p', seguidamente, y a modo de ejemplo, seleccionamos una de las columnas de una de las tablas dentro de nuestra base de datos. En nuestro caso la base de datos que hemos creado se llama 'fridex' y la tabla para una de las líneas se llama 'adentech'. Escribimos pues "SELECT potencia FROM fridex.adentech;"



```
mysql> SELECT potencia FROM fridex.adentech;
+-----------+
| potencia  |
+-----------+
|    0.1135 |
|    0.1145 |
|    0.1145 |
|    0.1165 |
|    0.1175 |
|    0.1185 |
|    0.1135 |
|    0.1115 |
|    0.1135 |
|    0.1145 |
|    0.1225 |
|  0.134999 |
|  0.131999 |
|    0.1155 |
|    0.1175 |
|    0.1235 |
|  0.134999 |
|  0.130999 |
```

Figura 5-14. Visualización de valores en base de datos MySQL en Ubuntu 16.04

REFERENCIAS

[1] <<"https://gruposinelec.com/eficiencia-energetica-en-el-marco-de-la-industria-4-0/">>

[2] <<"http://www.iet.es/wp-content/uploads/2013/03/REGLAMENTO-RBT-SEPT-2003.pdf">>

[3] <<"https://download.schneiderelectric.com/files?p_enDocType=User+guide&p_File_Name=EIO0000001432.07.pdf&p_Doc_Ref=EIO0000001432">>

[4] <<"http://www.plcdev.com/definition_of_a_plc">>

[5] <<"https://www.schneider-electric.com/resources/sites/SCHNEIDER_ELECTRIC/content/live/FAQS/299000/FA299371/es_ES/Smartlink%20IP%20User%20Manual%20(07-2016).pdf">>

[6] <<"https://www.schneider-electric.com/en/product-range-presentation/63626-powertag/#tabs-top">>

[7] <<"https://www.se.com/es/es/product-range-presentation/2226-ecostruxure%E2%84%A2-somachine/#tabs-top">>

[8] <<"https://nodered.org/">>

[9] <<"https://download.schneider-electric.com/files?p_enDocType=User+guide&p_File_Name=EIO0000003003.05.pdf&p_Doc_Ref=EIO0000003003">>

[10] <<"www.aie.cl">>

[11] <<"http://modbus.org/docs/Modbus_Application_Protocol_V1_1b3.pdf">>

[12] <<"http://wiki.opcfoundation.org/index.php/UA_Overview">>